

Pagination

Pagination A sophisticated and flexible pagination component designed to handle large datasets with intelligent page number display, multiple visual variants, and comprehensive accessibility features. Perfect for data tables, search results, content lists, and any interface requiring efficient navigation through paginated content. Uses the ava-pagination-controls selector for integration.

How to use ■ import { AavaPaginationControlsComponent } from "@aava/play-core" ; Basic

Usage ■ The most basic implementation with default settings and intelligent page number display.

Basic Features ■ Smart Ellipsis : Automatically shows ellipsis for large page counts Current Page

Highlighting : Active page is visually distinguished Navigation Buttons : Previous/Next buttons with

proper disabled states Responsive Design : Adapts to different screen sizes Intelligent Truncation

: Shows relevant pages around current selection Size Variants ■ Five size variants to

accommodate different interface densities and visual hierarchy requirements. Available Sizes ■

XSmall - Very compact size for minimal interfaces (12px font) Small - Compact size for minimal

content and dense interfaces (14px font) Medium - Standard size for most pagination scenarios

(16px font, default) Large - Prominent size for important content (20px font) XLarge - Very

prominent size for high-visibility interfaces (24px font) Size Features ■ Responsive Typography :

Font sizes scale appropriately with size variants Consistent Spacing : Button sizes and gaps scale

proportionally Touch Targets : All sizes maintain minimum 44px touch target requirements Visual

Hierarchy : Larger sizes provide better emphasis for important pagination Advanced Features ■

Icon-Only Navigation ■ The iconOnly property allows navigation buttons to display only icons

without text labels, creating a cleaner, more compact design. Clear Styling ■ The clear property

(used internally by unfilled variants) provides transparent button backgrounds for modern,

minimalist designs. Rounded Styling ■ The rounded property applies pill-shaped styling to page

number buttons for a softer, more modern appearance. Page Info Variants ■ Text-based

pagination variants that display current page information instead of page numbers. Page Info

Features ■ Text Display : Shows "Page X of Y" format Compact Design : Minimal space

requirements Clear Navigation : Previous/Next buttons for sequential navigation Accessibility :

Screen reader friendly with descriptive text Two Layouts : Full-width and centered variants

available Accessibility ■ Built-in accessibility features ensuring WCAG compliance and inclusive

user experience. Accessibility Features ■ Keyboard Navigation : Full Tab, Arrow key, Enter, and

Space support ARIA Labels : Proper labels for navigation buttons and page numbers Screen

Reader Support : Descriptive announcements for page changes Focus Management : Clear visual

focus indicators High Contrast : Enhanced visibility in high contrast mode Reduced Motion :

Respects user motion preferences API Reference ■ Inputs ■ Property Type Default Description

currentPage number 1 Current active page number totalPages number 10 Total number of pages

available type 'basic' | 'unfilled' | 'basicunfilled' | 'pageinfo' | 'pageinfilled' 'basic' Visual variant of

the pagination component showNavigationButtons boolean true Whether to show Previous/Next

navigationButtons rounded boolean false Whether to apply rounded styling to page number buttons iconOnly boolean false Whether to show only icons without text labels on navigation buttons size 'xs' | 'sm' | 'md' | 'lg' | 'xl' | 'md' Size variant of the pagination component customStyles Record<string, string> {} Custom CSS styles to apply to the pagination container Outputs ■ Event Type Description pageChange EventEmitter<number> Emitted when user navigates to a different page Methods ■ | Method | Parameters | Return Type | Description |

| ----- | ----- | ----- | ----- |

||

| nextPage() | - | void | Navigate to the next page |

| prevPage() | - | void | Navigate to the previous page |

| getFontSize() | - | string | Get font size based on current size prop |

| trackByPage(index, page) | index: number, page: number \| string | string | TrackBy function for ngFor optimization | Computed Properties ■ Property Type Description pages (number | string)[] Array of page numbers and ellipsis to display size 'xs' | 'sm' | 'md' | 'lg' | 'xl' Current size variant of the component CSS Custom Properties ■ | Property | Default | Description |

| ----- | ----- | ----- | ----- |

||

| --pagination-item-text | Dynamic | Text color for page numbers |

| --pagination-item-disabled-text | Dynamic | Text color for disabled elements |

||

| --pagination-container-gap | Dynamic | Gap between pagination elements |

| --pagination-page-label-color | Dynamic | Color for page label text |

| --pagination-page-label-font-family | Dynamic | Font family for page labels |

| --pagination-page-label-font-weight | Dynamic | Font weight for page labels |

| --pagination-page-label-line-height | Dynamic | Line height for page labels |

| --pagination-page-label-gap | Dynamic | Gap between page label elements | Accessibility Guidelines ■ Keyboard Navigation ■ Tab : Navigate to pagination and move between interactive elements Arrow Left/Right : Navigate between page numbers (when focused) Enter : Activate focused page number or navigation button Space : Activate focused page number or navigation button Home : Jump to first page (when supported) End : Jump to last page (when supported) Screen Reader Support ■ Use descriptive labels that clearly indicate pagination purpose Provide context about total number of pages and current position Announce page changes and navigation actions Include current page information in accessible descriptions Use appropriate heading levels for pagination sections Visual Design ■ Maintain sufficient color contrast (4.5:1 minimum) for all states Provide clear focus indicators on interactive elements Ensure page number buttons meet minimum touch target size (44px) Use consistent visual hierarchy across all variants Support high contrast and reduced motion preferences Best Practices ■ Design Guidelines ■ Appropriate Variant Selection : Choose variants based on dataset size and user needs Consistent Spacing : Maintain uniform spacing between pagination elements Clear Visual Hierarchy : Use proper

contrast and sizing for current page indication

Responsive Design : Ensure pagination works well on all screen sizes

Loading States : Consider loading indicators during page transitions

Performance

- **Efficient Rendering :** Use OnPush change detection for optimal performance

Event Handling : Debounce rapid page changes if triggered programmatically

Memory Management : Clean up event listeners and subscriptions

Bundle Optimization : Import only needed variants to minimize bundle size

Virtual Scrolling : Consider virtual scrolling for very large datasets

User Experience

- **Intuitive Navigation :** Make it easy to jump to first/last pages
- Page Size Options :** Consider adding page size selection for data tables
- Breadcrumb Integration :** Use pagination with breadcrumbs for complex navigation
- Search Integration :** Combine pagination with search functionality

URL State : Sync pagination state with URL parameters for bookmarking

Implementation Considerations

- **State Management :** Properly manage pagination state in your application
- Data Fetching :** Implement efficient data loading for each page
- Error Handling :** Handle edge cases like invalid page numbers
- Caching :** Cache previously visited pages for better performance
- Analytics :** Track pagination usage for user behavior insights
- Accessibility Implementation**
- **Semantic HTML :** Use proper HTML structure for pagination elements

ARIA Attributes : Implement appropriate ARIA labels and roles

Focus Management : Ensure logical tab order and focus indicators

Screen Reader Testing : Test with actual screen readers

Keyboard Testing : Verify complete keyboard navigation flow

Technical Notes

- **Component Architecture**

The pagination component uses a modern Angular architecture with:

PaginationControlsComponent - Main component with multiple variants

ButtonComponent integration for consistent button styling

OnPush change detection for optimal performance

ViewEncapsulation.None for global CSS variable access

MUI-Style Pagination Logic

- The component implements Material-UI style pagination with intelligent page number display:

Smart Ellipsis : Automatically shows ellipsis for large page counts

Context-Aware Display : Shows relevant pages around current selection

Optimal Truncation : Balances information density with usability

Responsive Behavior : Adapts to different total page counts

Size System

- The component provides a comprehensive size system:

Font Scaling : Each size maps to specific pixel values (12px to 24px)

Button Scaling : Button dimensions scale proportionally with font size

Touch Targets : All sizes maintain minimum 44px touch target requirements

CSS Variables : Uses semantic CSS variables for consistent theming

Variant System

- The component supports multiple visual variants:

Basic Variants : Traditional pagination with filled styling

Unfilled Variants : Modern design with transparent backgrounds

Page Info Variants : Text-based alternatives to page numbers

Icon-Only Options : Clean navigation without text labels

```
<aava-pagination-controls
  [type]="'basic'"
  [currentPage]="currentPage"
  [totalPages]="totalPages"
  [showNavigationButtons]="true"
  (pageChange)="onPageChange($event)"
  [rounded]="true"
>
</aava-pagination-controls>
```

```
currentPage = 1;
totalPages = 20;

onPageChange(page: number): void {
  this.currentPage = page;
}
```

```

<aava-pagination-controls
  [type]="'basic'"
  [currentPage]="basicPage"
  [totalPages]="10"
  [size]="'xl'"
  (pageChange)="onPageChange($event)"
></aava-pagination-controls>
<aava-pagination-controls
  [type]="'basic'"
  [currentPage]="basicPage"
  [totalPages]="10"
  [size]="'lg'"
  (pageChange)="onPageChange($event)"
></aava-pagination-controls>
<aava-pagination-controls
  [type]="'basic'"
  [currentPage]="basicPage"
  [totalPages]="10"
  [size]="'md'"
  (pageChange)="onPageChange($event)"
></aava-pagination-controls>
<aava-pagination-controls
  [type]="'basic'"
  [currentPage]="basicPage"
  [totalPages]="10"
  [size]="'sm'"
  (pageChange)="onPageChange($event)"
></aava-pagination-controls>
<aava-pagination-controls
  [type]="'basic'"
  [currentPage]="basicPage"
  [totalPages]="10"
  [size]="'xs'"
  (pageChange)="onPageChange($event)"
></aava-pagination-controls>

```

```

basicPage = 1;

onPageChange(page: number): void {
  this.basicPage = page;
}

```

■ No code found

■ No code found